RL-TR-92-34
In-House Report
February 1992

FOREIGN SURVEILLANCE RADAR UPGRADE ANALYSIS

Steven D. Farr

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

Rome Laboratory
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

This report has been reviewed by the Rome Laboratory Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RL-TR-92-34 has been reviewed and is approved for publication.

APPROVED: Thadeus J. Domurat

THADEUS J. DOMURAT, Chief
Signal Intelligence Division

FOR THE COMMANDER: Garry W. Barringer

GARRY W. BARRINGER
Technical Director
Intelligence & Reconnaissance Directorate

If your address has changed or if you wish to be removed from the Rome Laboratory mailing list, or if the addressee is no longer employed by your organization, please notify RL (IRAE), Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	February 1992	In-House Dec 90 - Dec 91

4. TITLE AND SUBTITLE
FOREIGN SURVEILLANCE RADAR UPGRADE ANALYSIS

5. FUNDING NUMBERS
PE - 62702F
PR - 4594
TA - 15
WU - I5

6. AUTHOR(S)
Steven D. Farr

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Rome Laboratory (IRAE)
Griffiss AFB NY 13441-5700

8. PERFORMING ORGANIZATION REPORT NUMBER
RL-TR-92-34

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Rome Laboratory (IRAE)
Griffiss AFB NY 13441-5700

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Rome Laboratory Project Engineer: Steven D. Farr/IRAE/(315) 330-4518

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)
This report details an example of the work required to predict the functionality of future systems based on improvement in computer technology. Areas of analysis include CPU load, bus load, memory load, I/O load, target report delay and probability of target loss associated with the upgraded radar. The analysis scenario provides an example of how queuing theory and probability can be applied for the purpose of assessing future threats.

14. SUBJECT TERMS
Queue Analysis, Probability & Statistics, Distributed Systems

15. NUMBER OF PAGES
46

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR

TABLE OF CONTENTS

Introduction 2
System Architecture 2
    Physical Structure 2
System Specifications 4
    CPU Performance Specifications 4
    Data System Messages 5
System Analysis 6
    Target Distribution Analysis 6
    Global Memory Design 7
    Radar Control Processing 9
        Pseudo-code for Radar Control Processing 9
        Timing Requirements 10
    Signal Processor 13
        SP Queue Analysis 14
    Target Processing 14
        Pseudo-code for Target Processing 15
        Timing Requirements 16
    Communication Processing 20
        Display Queue Analysis 20
        Modem Queue Analysis 21
        Pseudo-code for Communication Processing 22
        Timing Requirements 23
    Bus Loading 27
Summary of Analysis Results 29
Final Thoughts 30
Addendum
    M/D/1/n Queue Analysis A1
    M/D/1/n Program Output B1

Introduction

Assessment of foreign technology and its potential as a threat to the United States and its allies is often a problem of anticipating new developments. Development of U.S. weapon systems is often driven by a expected foreign threat as opposed to a direct result of an observed threat. For this reason, thorough analysis of potential system upgrades must be performed by intelligence analysts. An example of such an analysis would be the new performance characteristics of a surveillance radar in response to new CPUs. In a scenario such as this, the software subsystem is fairly well understood so the job of the analyst is to ensure that the software ported to the new computers will have adequate processing resources.

This report details the analysis scenario described above by assessing the CPU load, bus load, memory load, I/O load, target report delay and probability of target loss associated with the upgraded radar.

System Architecture

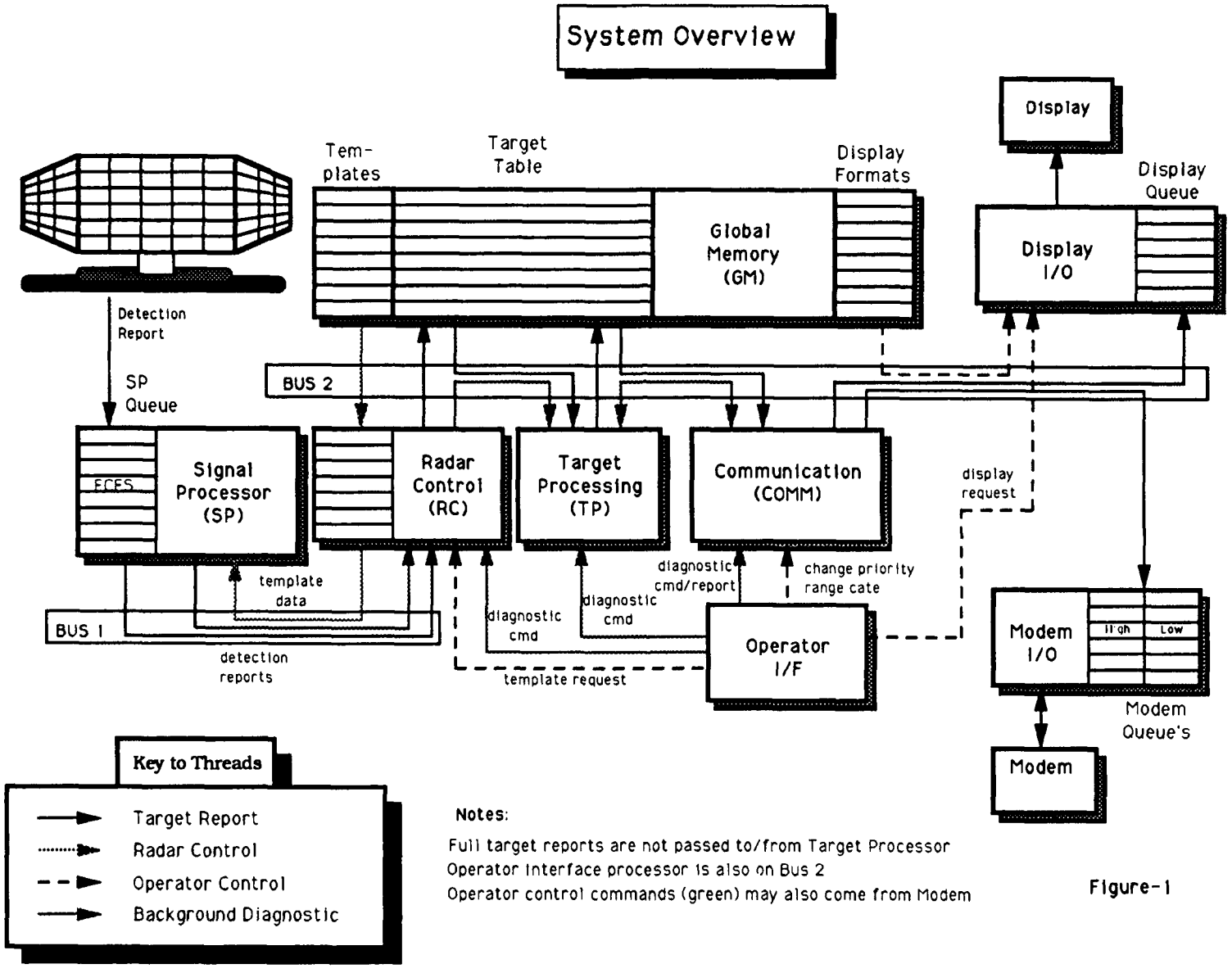
Physical Structure

The radar data system is comprised of several components including five CPUs, global and local memories, I/O handlers, a display, a modem, buses, and the radar itself. This particular system is fictitious, but based in-part on a U.S. surveillance radar.

The radar is mechanically scanned in azimuth at a rate of 6 seconds per revolution. Each revolution consists of 200 segments, each of which contains one elevation scan. An elevation scan results from phase steering the antenna array and has 15 beams from 0 to 20 degrees in elevation. From these parameters it is easily seen that one elevation scan occurs every 0.03 seconds.

The system inter-connectivity is shown in Figure 1. The four principle functions; Radar Control, Target Processing, Communications and Display, are each handled by a separate CPU which passes messages and data through the system. Each of these computers and the I/O handlers have their own local memory. User control is maintained through a Operator Interface (I/F) which communicates directly with each of the other functions as opposed to passing messages through global memory.

Figure 1 also shows the data processing threads for this system. A thread is defined as the processing initiated by some specific recurring event. The target report thread is the processing associated with the handling of each target detection report. This thread is initialized by the arrival of a target into the First-Come First-Served queue inside the signal processor (SP). If the radar control (RC) function doesn't access target reports from SP in a timely fashion, the input queue will overflow. Queuing analysis for every queue in this system must therefore be



Notes:

Full target reports are not passed to/from Target Processor
Operator Interface processor is also on Bus 2
Operator control commands (green) may also come from Modem

Figure-1

performed since it is assumed that for cost and real estate' reasons, queues will be kept at a minimally acceptable length.

System Specifications

The required specifications for system performance during both nominal and peak scenarios are identified in Table 1. These specs are usually derived from what opposition forces anticipate facing in the way of U.S. and allied forces. False alarms are processed identically to actual detection reports, and are made available to the target processor for analysis. If the target processor does not eliminate the false report, it is then sent to the display and out the modem link. If the original system was capable of eliminating 50% of the false targets through target processing, one can therefore assume that same rate for the new system would be appropriate and thus all analyses will assume 200 and 400 false alarms out of the signal processor for nominal and peak loads, respectively.

Table 1 - System Specifications

Specification	Nominal Load	Peak Load	Comments
Target Load	500	1200	reports/scan
False Alarms	100	200	reports out/scan
Display Request & Operator Commands	1	1	requests/minute
Priority Filters	2 priority 1 all else priority 0	2 priority 1 all else priority 0	defined by operator chosen sectors
Background Diagnostics	once / 10scans	none	
Probability of Target Report Loss	1 %	1 %	The data system is allowed to lose 1 out of 100 targets
CPU spare	50 %	40 %	measure by process time required to complete one elevation scan

CPU Performance Specifications

As mentioned earlier, the basis for the system upgrade is the installation of new computers to handle processing requirements. Advances in foreign computing capability is closely monitored by U.S. intelligence organizations. Whether the new computers were developed through foreign research and development or by the acquisition of Western technology is not pertinent to this discussion since only fictitious CPU performance specifications are used. Table 2 contains the performance criteria for the new computers.

Table 2 - CPU performance specifications

Processing Load	CPU time (msec)	comment
RC get templates from GM	10 msec/template + data transfer	as often as 1/min
RC send templates to SP	data transfer	once per elevation scan
TP no target load background	100 msec/scan	evenly distributed throughout the scan (one revolution)
TP coordinate estimation	2.0 msec/report	converts elevation angle to altitude then performs table look-up to correct for atmospheric disturbance
COMM no target load background	150 msec/scan	evenly distributed throughout the scan
COMM data transfer	cmds from Oper I/F, modem + rpts from target table + reports to display + rpts out modem	Oper I/F and modem commands arrive 1/minute maximum
Radar Diagnostics	200 msec/diag report in TP + transfer from COMM to display	
Self diagnostics	3 seconds to complete	distributed throughout 10 scans

All other CPU performance specifications - RC handling of target reports, TP data editing, TP table management, and COMM retrieval of target reports will be estimated from pseudo code.

Data System Messages

The bulk of the processing load will be due to messages being transferred around the system. Message transfer times are determined from the summation of the setup times and the time required to send the data over the bus. Data transmittal with this system is quite simple. Data transmitted over the system bus requires setup of 500 µsec for RC, TP, GM, OIF and COMM and transfers at a rate of 200 bytes per msec. This transfer rate also applies to the display subsystem, however setup time is 700 µsec. The modem also requires a 700 µsec setup time, but can only transfer data at a rate of 1.2 bytes per msec.

Target reports are 10 bytes long and consist of a range, azimuth, elevation, time, and type for each target. Despite the use of azimuth hashing[1], azimuth is stored because azimuth angles are measured more accurately than gross segment hashing angles.

[1] the hashing function takes an azimuth and returns an index into the target table (0 - 199).

Table 3 - Data System Messages

Messages	Min Length bytes	Max Length bytes	Setup Time msec	Data Rate bytes/msec
SP Detection Report	10	10	0.3	500
Detection Report to GM	10	10	0.5	200
Template from RC to SP	400	2000	0.3	500
Target Reports to Display	25	35	0.7	200
Target Reports to Modem	10	10	0.7	1.2
Operator commands	5	5	0.5	200
Formats from GM to Display	1200	1500	0.7	200
Template from GM to RC	400	2000	0.5	200
Diagnostic Reports to TP	100	250	0.5	200
Diagnostic Reports to COMM	100	250	0.5	200
Diagnostic Reports to Display	100	250	0.7	200

System Analysis

Target Distribution Analysis

One implementation of an azimuth hashing scheme partitions the area of radar coverage into 200 sectors of equal area as shown in Figure-2; therefore the probability that any one target is in any given sector is 0.005 (1/200). Furthermore, the probability that any n of the 1600 (1200 real, 400 false) targets in the coverage area lie in a given sector is determined by the binomial distribution:

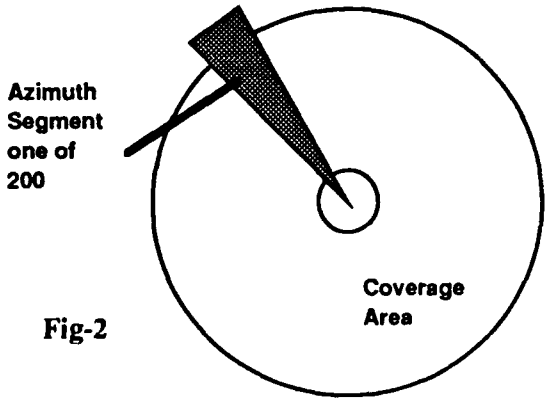


Fig-2

$$P\{n \text{ targets in sector}\} = \binom{N}{n} p^n (1-p)^{N-n}$$

Which can be estimated by the Poisson Distribution since N>>p.

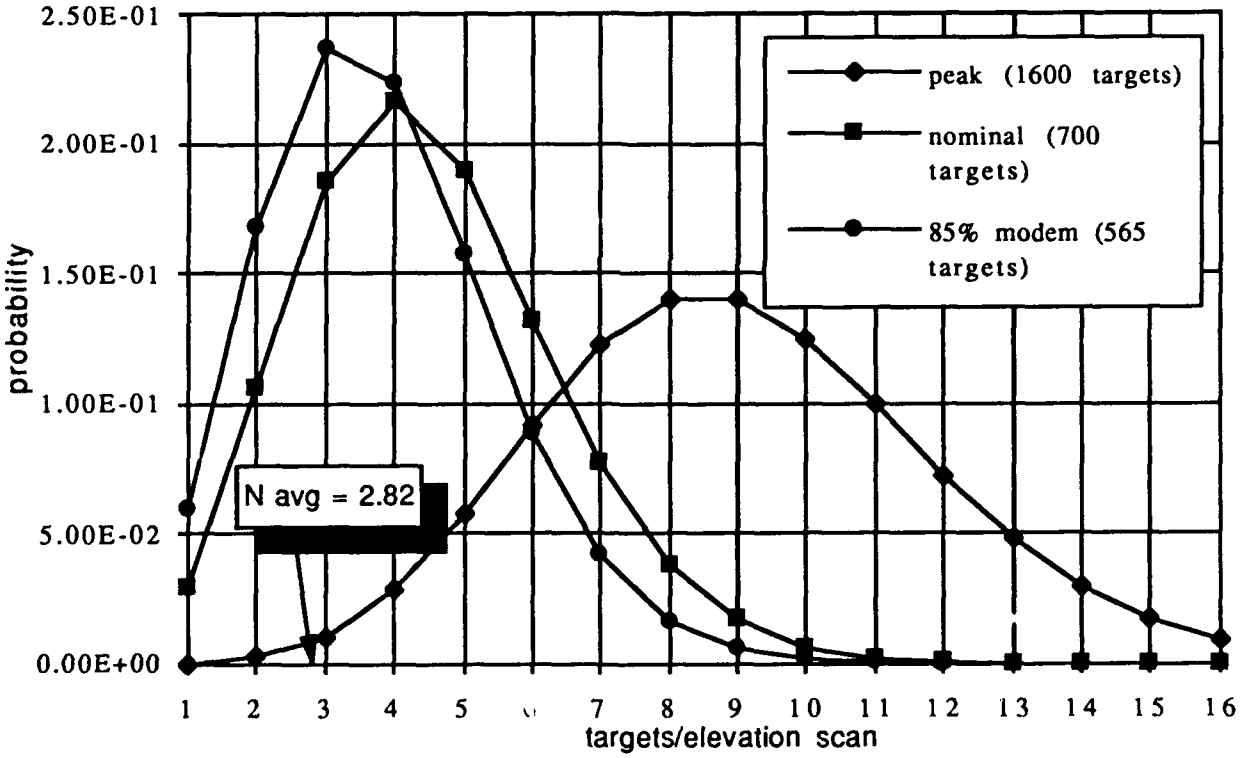
$$P(n \text{ targets in sector}) = e^{-Np} \frac{(Np)^n}{n!}$$

A family or mass functions, each of which is dependent upon the number of targets in coverage, can be used to describe this system. Figure 3 shows the function for peak, nominal and a third load which is representative of the system when the modem is operating at 85% capacity. The third curve which corresponds to 565 targets in coverage will also be used as a means for estimating CPU load and represents a typical day-to-day load. The 565 targets were derived from:

$$\frac{\text{sweep time}}{\text{service time}}\ 200 \text{ seg } (0.85) = \frac{0.03 \text{ sec}}{\frac{10 \text{ bytes}}{1200 \text{ bytes/sec}} + 0.0007 \text{ sec}} (200 \text{ seg}) (0.85)$$

$$= 564.57 \text{ reports/scan}$$

Figure 3 - Azimuth List Lengt



Global Memory Design

A likely scenario is that target reports are stored in global memory by the implementation of a circular buffer. The use of a circular buffer would eliminate the need to purge the table for each scan by arranging the memory so that the scan for true North always begins at the same position as shown in Figure 4.

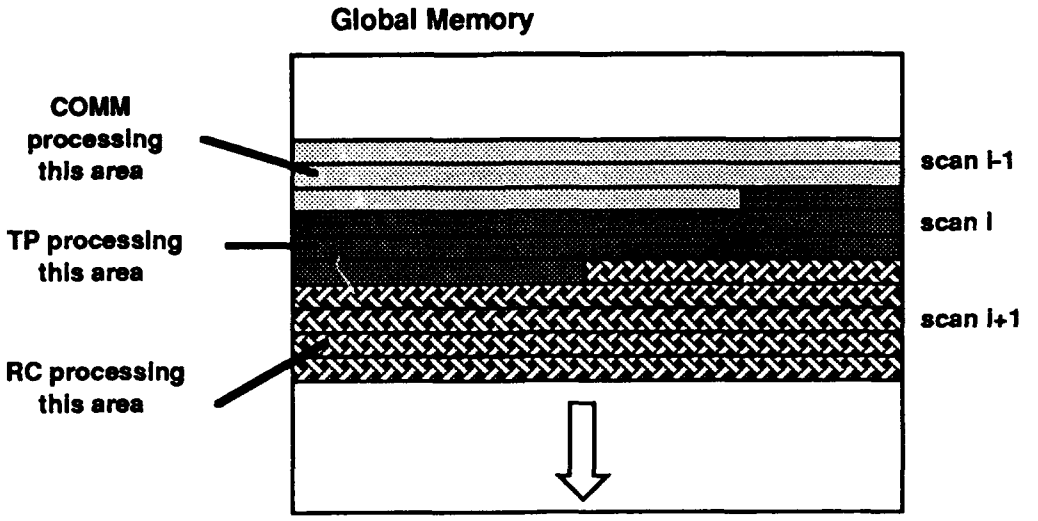


Fig-4

Access to the target data can be controlled by the utilization of semaphores so that RC, TP and COMM do not experience race conditions[2]. When the antenna begins to scan, RC will set the first semaphore (S_1) to 0 to indicate that it only has access to the area of global memory designated for the first azimuth-elevation scan. When RC finishes an elevation scan, it increments the semaphore, and, if possible, begins to process the next sector. TP then builds the linked data structure (represented by L_i in Figure-5), increments the semaphore and, if RC has completed, moves on to the next scan. The COMM processor then performs its function and sets the semaphore back to 0.

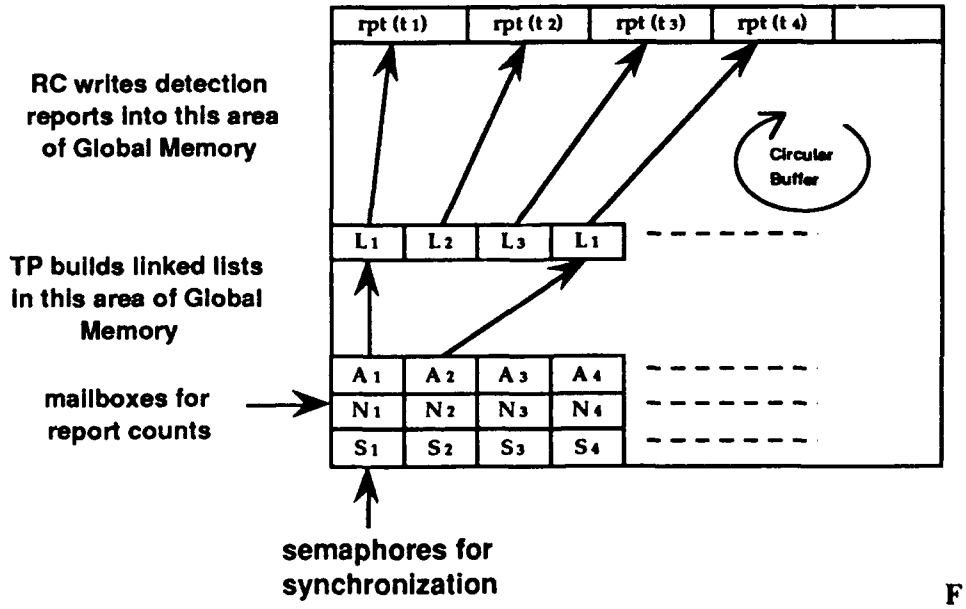


Fig-5

[2]when two or more processes are reading or writing some shared data and the final result depends on who runs precisely when.

The three processors sequentially perform their respective functions with the target data. So long as each processor can perform its function in less than the time required to perform one elevation scan, then all three processors will move in lock step. In the event that one of the processors require more than 30 msec (one elevation scan time) to perform its tasking, then the processor will lag (in terms of which elevation scan it is processing) behind the others, but will eventually catch up so long as the average list length can be handled within 30 msec. If this is not possible, the system will lock up and target data will be lost until the semaphore allowing RC to continue processing is set.

Radar Control Processing

The primary function of the Radar Control processor is moving target detection reports from the signal processor queue to the circular buffer in global memory. RC accomplishes this by reading the reports from the SP buffer to its own local memory. RC first places a pointer in the A_j location which contains the address of the first report of the j^{th} elevation scan and then sequentially writes reports to GM. Before proceeding to the next elevation scan, RC writes the number of reports, N_j, to global memory. Prior to accessing detection reports, this processor must supply templates to SP which it retrieves from global memory for waveforms and steering angles for each beam to the Signal Processor. Other overhead functions include the creation and handling of diagnostic reports.

Pseudo-code for Radar Control Processing

Pseudo-code for the three main processing functions (RC, TP, and COMM) are written for purposes of estimating report handling execution times. The format of the pseudo-code is akin to C (psuedo-C?). Comments are used to describe data transfer only. Execution times for all processors are based on 2 μsec access to GM and 0.85 μsec access to local memory. Each High Level Instruction takes 4 x 0.85 = 3.4 μsec plus any access time to global memory.

pseudo-code	comment	execution time (μsec)
IF (j == 0)		3.40
A_1 = 0;	/* RC local to GM */	5.40 * (1/200)
Next_Azimuth=0;		3.40 * (1/200)
{		
ELSE		
A_j = Next_Azimuth;	/* RC local to GM */	5.40 * (199/200)
counter=0;		3.40
WHILE (target report in SP queue) {		3.40 * N
	/* SP local to RC local*/	
rc.report(counter) = sp.report;		(0.3 + 10/500) * 1000 * N
counter++;		3.40 * N
}		
FOR(index=1; index<=counter; index++)	/* RC local to GM*/	3.40 * N

```
                                        /* RC local to GM */
   &gm.report+10  =  rc.report(index);                          (0.5 + 10/200) * 1000 * N
}
N_j=counter;                            /* RC local to GM */      5.40
Next_Azimuth  =  counter*10;                                      3.40
j++;    (modulo 200)                                              3.40
semaphore_j = 1;                                                  5.40

                                                                 ------
                                                                 880.2*N + 29.8
```

Timing Requirements

The CPU load for the RC processor consist of the summation of the times necessary to send the template to SP, perform table management, retrieve templates from GM when they have been requested by the operator, carry out background diagnostics, and perform diagnostic report handling. Of these functions, only the table management execution time varies with the number of targets to be processed. Using the 565 target load as a basis, processing times for RC and the probability that each particular processing time will be required are tabulated in Table 4.

$$\text{send template to SP} = 0.3 + \frac{2000}{500} = 4.3 \text{ msec}$$

$$\text{table management} = 0.8802 * N + 0.0298 = \text{time (msec)}$$

Although constant, the 4.3 msec required to transfer a template to SP is included because this function, unlike operator requests, must be performed for each elevation scan.

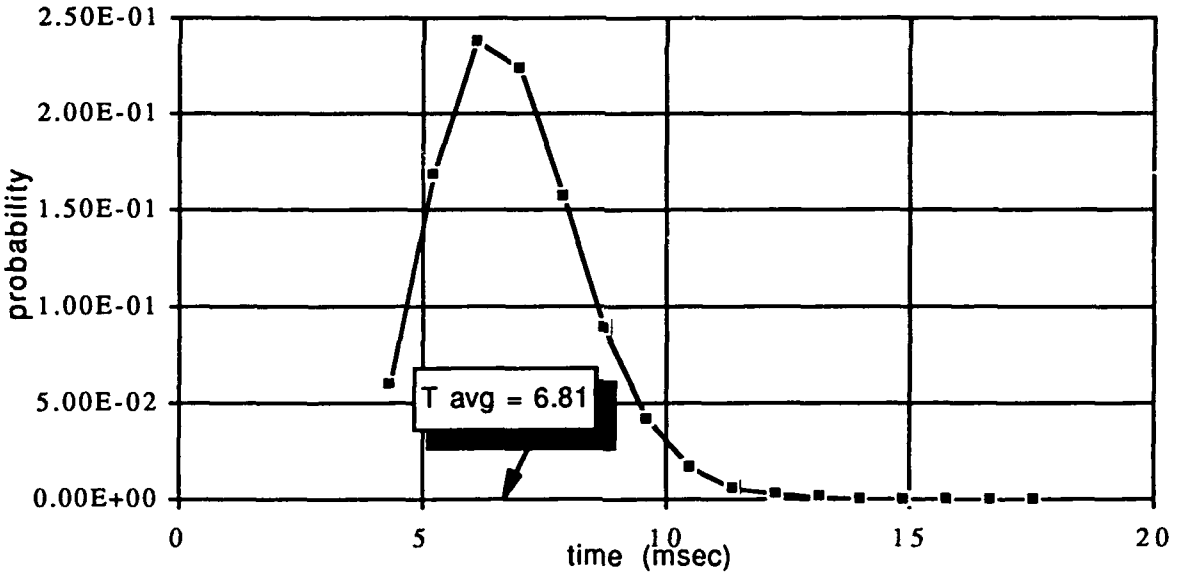
Table 4 - Radar Control Processing Times

Number of Reports	Exec. time[3] (msec)	P{list length=N}	Number of Reports	Exec. time (msec)	P{list length=N}
0	4.33	5.94E-02	8	11.37	5.94E-03
1	5.21	1.68E-01	9	12.25	1.86E-03
2	6.09	2.37E-01	10	13.13	5.26E-04
3	6.97	2.23E-01	11	14.01	1.35E-04
4	7.85	1.57E-01	12	14.89	3.18E-05
5	8.73	8.88E-02	13	15.77	6.90E-06
6	9.61	4.18E-02	14	16.65	1.39E-06
7	10.49	1.68E-02	15	17.53	2.62E-07

Plotting this data set shows that the average processing time of 6.81 msec will occur with a probability of approximately 0.225.

[3]Execution time = table management + sending template to SP (constant 4.3 msec).

Figure 6 - RC Processing Tim



During peak load, the only operator request that can be issued to this processor is for a template change (a request for a diagnostic report is not applicable because CPU diagnostics are typically not performed during peak load scenarios); therefore, any remaining CPU time can be allocated to this task. If the processor is to function with a 40% spare then:

send template to SP = 4.3 msec
table management = 880.2 * 8 + 29.8 = 7071.4 μsec = 7.1 msec
background diagnostics = 0.0.
operator request for diagnostic report = 0.0

remaining CPU time = (30 * 0.6) - (4.3 + 7.1) = 6.6 msec/seg

and

new template processing = get operator request + get template from GM

$$= \left(0.5 + \frac{5}{200}\right) + \left(10 + \left(0.5 + \frac{2000}{200}\right)\right)$$

$$= 0.525 + 20.525 = 21.025 \text{ msec}$$

If the 6.6 msec is used to handle a request for a new template, then the operator can expect a response for the new template request within:

$$30.0 * \left(\frac{21.025}{6.6}\right) = 95.6 \text{ msec before new template arrives at SP}$$

During nominal load processing, two different operator requests can be serviced; the request for a new template and a request for a diagnostic

report. These requests are to be handled during the remaining CPU time after RC has performed its principle tasks.

send template to SP = 4.3 msec

table management = 880.2 * 3.5 + 29.8 = 3110.5 μsec = 3.1 msec

background diagnostics = 3 seconds / 10 scans = 1.5 msec

Recalling that the processor operates at 50% spare during nominal loading:

remaining CPU time = (30 * 0.50) - (4.3 + 3.1 + 1.5) = 6.1 msec

and

diagnostic report processing = get operator request + get SP's report + send both reports to TP

$$= \left(0.5 + \frac{5}{200}\right) + \left(0.3 + \frac{250}{500}\right) + 2\left(0.5 + \frac{250}{200}\right)$$

$$= 0.525 + 0.8 + 3.5 = 4.8 \text{ msec}$$

If 6.1 msec are used to handle either operator request, then the expected response time for each would be:

$$30.0 * \left(\frac{21.025}{6.1}\right) = 103.4 \text{ msec to send new template to SP}$$

diagnostic reports can be expected to leave RC during one segment since 4.8 < 6.1

It is interesting to note that if the user specification with respect to CPU spare is maintained with this design which allows for new templates during peak activity, the templates will arrive at SP sooner during peak conditions.

With this design, there are three conditions which impact the CPU spare of this processor (1) when operating at 50% capacity and the number of targets become so large that the operator request for a new template cannot be handled within 1 minute, (2) when operating at 60% and the number of targets becomes so large that a 40% spare cannot be maintained and (3) number of targets becomes so great that it is not possible to accept targets and still get the template over to the SP.

To meet (1) the following condition must be held:

$$\text{CPU spare} \geq \frac{1 \text{ minute}}{21.025 \text{ msec}}$$

$$15 - (4.3 + 1.5 + 0.8802 * N + 0.0298) \geq 0.00035 \text{ msec} \Rightarrow N \leq 10.4$$

This calculation shows that even if the average number of targets is 10.4 (corresponding to 2082 targets) 50% CPU load can be maintained on *average* throughout the sweep, but there may be conditions in one

elevation scan where this can't be upheld. As a worse case, the probability of more than 10 targets in one elevation scan during peak load is:

$$P(>\ 10 \text{ targets inside sector}) = 1 - \sum_{j=0}^{10} P\{j \text{ targets}\}$$

$$= 0.183$$

At above 10 targets, RC will operate at 40%. This spare can be maintained as long as the list length is less than 14 as determined by:

$$18 - (4.3 + 1.5 + 0.8802 * N + 0.0298) \geq 0.00035 \text{ msec} \quad \Rightarrow \quad N \leq 13.8$$

The probability of this occurring is:

$$P(>\ 13 \text{ targets inside sector}) = 1 - \sum_{j=0}^{13} P\{j \text{ targets}\}$$

$$= 0.034$$

Looking at the extreme case, there may be a problem with the system backing up if any of the CPUs cannot finish their processing within one elevation scan (i.e., 30 msec). This is particularly critical for the case of RC. If RC cannot remove reports from the SP queue, they will not be processed irregardless of whether or not TP and COMM whould have been free to handle them. It is therefore imperative that RC never lag. To check this, the maximum number of targets that RC can handle in one azimuth sweep and still send a template to SP is given by:

$$\frac{\text{sweep time}}{\text{service time}} \times 200 =$$

$$\frac{0.03 \text{ seconds} - 0.0043 \text{ seconds}}{0.00091 \text{seconds/target}} \times 200 = 5648 \text{ targets} < 1600$$

If the user ever wishes to change the specifications to handle more targets in coverage, this processor is capable of managing up to 5600 reports. I wouldn't recommend that this be done, but these figures do provide a ceiling for this processor. For the purpose of this analysis, it is safe to say that the new computer is well suited for RC since it was shown that this processor can maintain a 50% spare and still handle in excess of 2000 targets.

Signal Processor

As shown in Figure 1, the Signal Processor lies between the radar itself and the Radar Control processor. One purpose of this processor is to

receive target reports and queue them up until RC is ready to accept them. The SP processor also sends template data to the radar at a rate of one per elevation scan. Templates are received from RC at the same rate. This arrangement is established because SP does not typically have sufficient local memory to store an entire azimuth scan's worth of template data. SP also performs self diagnostics that must be passed in a report form to TP through RC.

SP Queue Analysis

Although SP lies sequentially ahead of RC (in terms of report handling), the expected service time from RC analysis is needed to analyze the SP queue. For the purpose of this report, the assumption is made that SP places detection reports into the SP queue with Poisson arrival statistics; therefore the arrival rate at peak conditions is given by:

$$P\{k\} = e^{-\lambda t}\frac{(\lambda t)^k}{k!}$$

where

$$\lambda = (1200 \text{ reports/scan} + 400 \text{ failures/scan}) \frac{1}{6 \text{ seconds}} = 267$$

The service time associated with RC, t, is equal to the sum of the transfer time from SP to RC, the transfer time from RC to GM and the overhead associated with RC's handling of the report. These three items are all inherent to the pseudo-code execution time developed for RC; therefore the service time for 1 detection report is:

$$\text{service time} = 880.2 * 1 + 29.8 = 910 \ \mu\text{sec} = 0.00091 \text{ seconds}$$

By realizing that this system is modeled with a constant service time, the M/D/1/n queue analysis method attached as an annex to this report can be used to determine a suitable queue size given a desired probability of overflow equal to 0.25% and λt equal to 0.24 (267 x 0.00091). The steady state vector below shows that a queue size of 3 will satisfy the specifications.

$$V = \left[7.603 \times 10^{-1} \quad 2.062 \times 10^{-1} \quad 3.020 \times 10^{-2} \quad 3.061 \times 10^{-3} \quad 2.499 \times 10^{-4} \right]$$

Target Processing

The Target Data Processor is responsible for three major report handling functions; (1) weeding out duplicate reports from adjacent beams

and reports that have unlikely parameters[4], (2) estimating target altitude, and (3) storing the resultant report in a Global Memory-based target table.

Processing begins by retrieving the first report from this elevation scan by using the A_j pointer left by RC. TP then reads each of the N_j reports into its local memory and sorts them using the insertion sort routine. This sort algorithm is a function of N^2 as opposed to faster log(N) sort, but since the linked list is expected to be relatively short and already close to range order, it is not necessary to employ a more efficient sort function.

After the reports have been sorted by range, duplicate reports are eliminated simply by not linking in the redundant report as opposed to the more time consuming function of deleting the report from Global Memory. Duplicate reports are identified by comparing the range and elevation of the current report with that of the previous one. If the ranges and elevations are equivalent (within a reasonable tolerance), TP will treat the reports as redundant. Estimation of target altitude is based on elevation angle, correcting for atmospheric effects by use of a table look-up. This table is stored in global memory.

Pseudo-code for Target Processing

* Discussion of pseudo-coding practices are discussed in the Radar Control processing.

pseudo-code	comment	execution time
temp = $link_j$;	/* GM to TP local */	5.40
$link_j$ = $azimuth_j$;	/* GM to GM */	4.00
$azimuth_j$ = &$link_j$;	/* GM to GM */	4.00
FOR (index = 1; index <= N_j; index++) {	/* GM to TP local */	2.0 + 3.40 * N_j
	/* GM to TP local */	5.40 * N_j
temp.range(index) = gm.report.range(index);		
	/* GM to TP local */	5.40 * N_j
temp.elevation(index) = gm.report.elevation(index);		
}		
FOR (index = 1; index <= N_j; index ++)	/* GM to TP local */	2.0 + 3.40 * N_j
map(index) = index -1;		3.40 * N_j
insertion_sort(temp,map,N_j);		$1.7N_j^2$ + 22.1N_j - 6.8
FOR (index = 1; index <= N_j; index++) {	/* GM to TP local */	2.0 + 3.40 * N_j
if (temp.range(index) == prev_temp.range(index) &		3.40 * N_j
temp.elevation(index) == prev_temp.elevation(index))		
map(index) = -1;		0
}		
temp_link = $link_j$;	/* GM to TP local */	5.40
counter = 0;		3.40

[4]unlikely combinations of range, altitude and velocity.

FOR (index = 1; index <= N_j; index++) { /* GM to TP local */	2.0 + 3.40 * N_j
if (map(index) >= 0) {	3.40 * N_j
counter++;	3.40 * N_j
/* TP local to GM */	
$link_{counter}$ = temp_link + 10*map(index);	5.40 * N_j
}	
}	
N_1=counter; /* TP local to GM */	5.40
FOR (index = 1; index <= N_j; index++) { /* GM to TP local */	2.0 + .40 * N_j
prev_temp.range(index) = temp.range(index);	3.40 * N_j
prev_temp.elevation(index) = temp.elevation(index);	3.40 * N_j
}	
$semaphore_j$ = 2;	5.40
j++; (modulo 200)	3.40
}	
	$1.7N_j^2 + 70.3N_j + 45$

---

insertion_sort(data,map,N) {	3.40 * (N+2) + 3.40[5]
FOR (index = 1; index <= N; index++) {	3.40 * (N-1)
temp = data(index);	3.40 * (N-1)
temp_map = map(index);	3.40 * (N-1)
counter = index;	3.40 * (N-1)
while (data(counter-1) > temp) {	3.40 * (N-1) * N/8[6]
data(counter) = data(counter - 1);	3.40 * (N-1) * N/8
map(counter) = map(counter - 1);	3.40 * (N-1) * N/8
counter--;	3.40 * (N-1) * N/8
}	
data(counter) = temp;	3.40 * (N-1)
map(counter) = temp_map;	3.40 * (N-1)
}	
}	3.40
	$1.7N^2 + 22.1N - 6.8$

Timing Requirements

The Target processing load consists of table management, a no target load background, coordinate estimation, background diagnostics, and servicing an operator request for a diagnostic report. As with RC, only table management time varies with the number of targets to be processed. Again, the 565 target load is used as a basis for studying the processing times and their respective probability of occurrence. Table 5 and Figure 7 show the target table management load as a function of target reports.

---

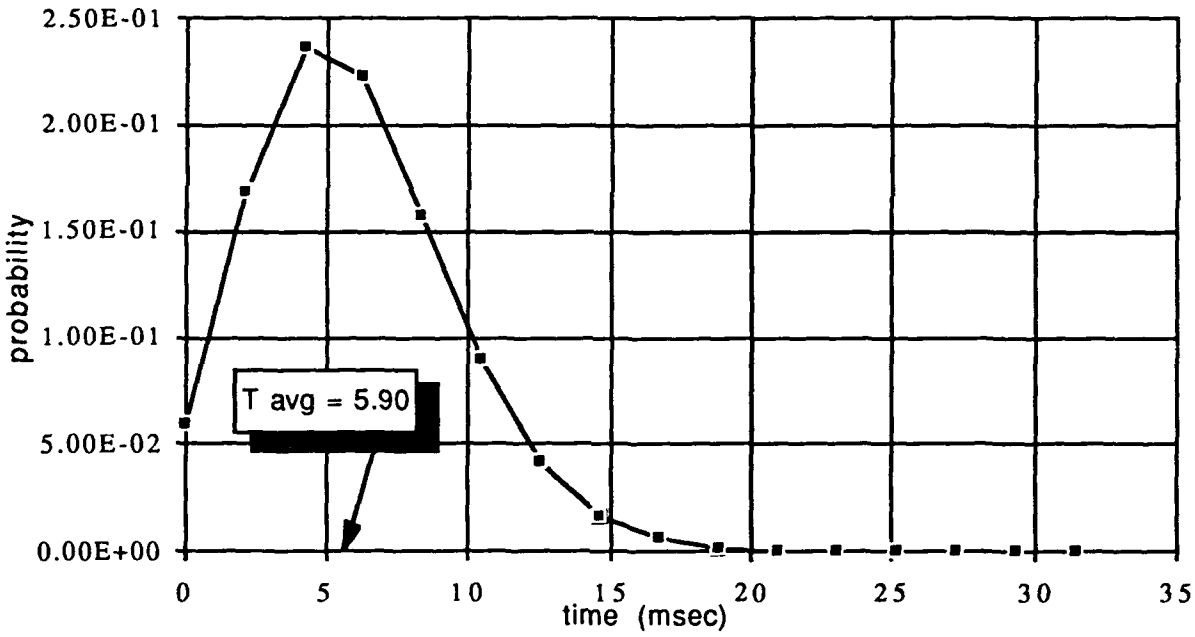
[5]The N+2 term is derived from the fact that N data values, the value of N and the address of map are placed on the stack.

[6]This loop is executed an average of N/8 times

Table 5 - Target Processing Times

Number of Reports	Exec. time (msec)[7]	P{list length=K}	Number of Reports	Exec. time (msec)	P{list length=K}
0	0.045	5.94E-02	8	16.72	5.94E-03
1	2.12	1.68E-01	9	18.82	1.86E-03
2	4.19	2.37E-01	10	20.92	5.26E-04
3	6.27	2.23E-01	11	23.02	1.35E-04
4	8.35	1.57E-01	12	25.13	3.18E-05
5	10.44	8.88E-02	13	27.25	6.90E-06
6	12.53	4.18E-02	14	29.36	1.39E-06
7	14.62	1.68E-02	15	31.48	2.62E-07

Figure 7 - TP Processing Tim



During peak conditions, TP is not required to process radar diagnostic reports. Total load during these conditions is expected to be 17.22 msec. and was determined by summing the following times:

table management = $1.7(8)^2 + 70.3(8) + 45 = 716.2$ μsec = 0.7162 msec

no target load background = $\frac{100 \text{ msec}}{200 \text{ seg}} = 0.5$ msec

[7]Execution time = table management + coordinate estimation

coordinate estimation = 2 * N = 2 * 8 = 16.0 msec

background diagnostics = 0.0

Since operator requests are not processed during peak conditions, it is only necessary to determine if the above processing time meets the 40% spare specification. The calculation below suggests that during peak conditions, specifications for an average CPU spare can be met and, although TP may lag on occasion, the entire azimuth scan processing can be performed in such a way that the radar will not lock up.

$(0.72 + 0.50 + 16.0) < 30.0 * .60 \quad \Rightarrow \quad 17.22 < 18.0$

During nominal load conditions, operator requests for diagnostic reports must be handled. Required processing times for nominal loads are:

table management = $1.7(3.5)^2 + 70.3(3.5) + 45 = 266.8$ μsec = 0.27 msec

no target load background = $\frac{100 \text{ msec}}{200 \text{ seg}} = 0.5$ msec

coordinate estimation = 2 * N = 2 * 3.5 = 7.0 msec

background diagnostics = $\frac{3 \text{ sec}}{10 \text{ scans}} = 1.5$ msec

Because the processor operates at 50% spare during nominal loading:

remaining CPU time = (30 * 0.50) - (0.27 + 0.5 + 7.0 + 1.5) = 5.73 msec

and

diagnostic report processing = get operator request + receive 3 reports (SP, RC, COMM) +200 msec overhead + send composite report to COMM

$$= \left(0.5 + \frac{5}{200}\right) + 3\left(0.5 + \frac{250}{200}\right) + 200 + \left(0.5 + \frac{5*250}{200}\right)$$

$$= 0.525 + 5.25 + 200 + 6.75 = 212.5 \text{ msec}$$

If 5.73 msec is used to handle the operator request for a diagnostic report, then the expected response time to send a composite report to COMM (discounting any time spent waiting for the other processors to send their diagnostic reports) would be:

30.0(212.5/5.73) = 1,112.6 msec

There are two conditions which impact the CPU spare of this processor (1) when operating at 50% capacity and the number of targets

become so large that the operator request for a diagnostic report cannot be handled within 1 minute and (2) when operating at 60% and the number of targets become so large that a 40% spare cannot be maintained.

To meet (1) the following condition must be held:

$$\text{CPU spare} \geq \frac{1 \text{ minute}}{212.8 \text{ msec}}$$

$$15 - (0.0017(N)^2 + 0.0703(N) + 0.045 + 2(N)) \geq 0.00355 \text{ msec} \quad \Rightarrow \quad N \leq 7.18$$

This calculation shows that above 7 targets, the TP CPU must switch to using 40% spare. Once switching to an 18 msec maximum (i.e, 40% spare), the CPU can handle up to:

$$0 \geq 18 - (0.0017(N)^2 + 0.0703(N) + 0.045 + 2(N)) \quad \Rightarrow \quad N \leq 8.61$$

This calculation shows that the CPU only operates above 40% spare when less than 9 targets are present in one elevation scan. Knowing that at peak where the average list length is 8, it is likely that 9 or more targets will be present. Specifically, this probability is:

$$P(> 8 \text{ targets inside sector}) = 1 - \sum_{j=0}^{8} P\{j \text{ targets}\}$$

$$= 0.407$$

The final extreme would be when TP cannot perform its processing in less than 100% utilization. For this to occur, the number of targets in one elevation scan would have to be:

$$0 \geq 30 - (0.0017(N)^2 + 0.0703(N) + 0.045 + 2(N)) \quad \Rightarrow \quad N \leq 14.3$$

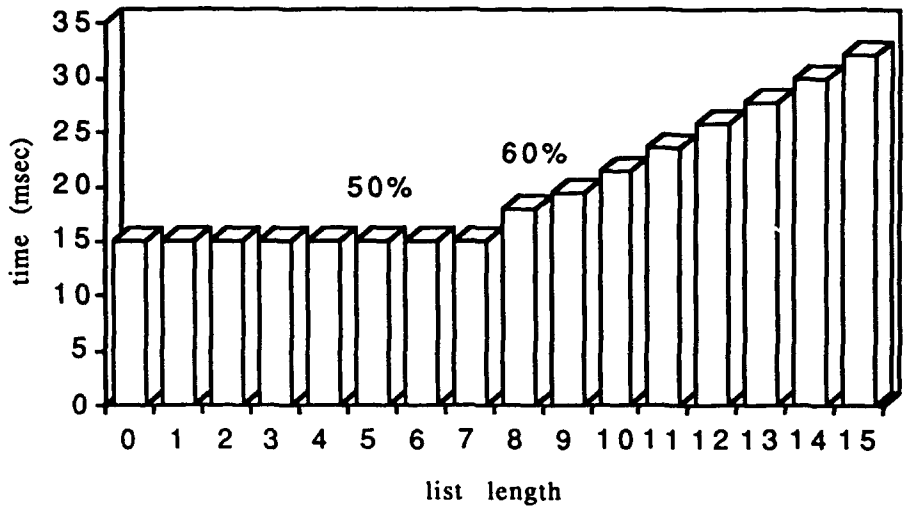
Therefore, if more than 14 targets appear in one elevation scan TP will lag. It will, however, catch up before the entire azimuth scan is complete since TP was shown to be able to handle an average of 8 targets/elevation scan. The probability of TP not being able to complete its processing during a 30 msec elevation scan during worse case conditions is:

$$P(> 14 \text{ targets inside sector}) = 1 - \sum_{j=0}^{14} P\{j \text{ targets}\}$$

$$= 0.017 \text{ or } 1.7\%$$

Figure 8 shows the fully loaded TP processing times. When the CPU is processing operator requests, it will be pushed to the maximum CPU spare allowable by specification. After a list length of 8, TP will operate at less than 40% spare.

Figure 8 - Full TP processing loa



Based on these calculations, the analyst can conclude that TP can meet the specification if the new computers are used, but only by a narrow margin.

Communication Processing

The COMM processor is tasked with sending data to the modem and display. The target data for each elevation scan is read into COMM's local memory and immediately sent to the display queue. However, the COMM modem processing is not as simple because the modem queue is apt to back up during greater than nominal conditions. If the target data cannot be placed into the modem queue, the report's reference data (A_j, N_j and Segment Number) is placed into a backup array so that COMM knows where to find the data when the modem queue slots become available. If data from any segment is still waiting to be sent to the modem when that segment comes around again, the old data will be lost. In other words, data marked as high priority must be placed into the modem queue within 6 seconds, or else it will be superseded by the updated scan data for that segment.

Display Queue Analysis

Analyzing the display queue as a worse case scenario (i.e., service time is fixed a 2 msec), the maximum number of targets that can be serviced during the 0.03 second elevation scan is:

$$\frac{30 \text{ msec/scan}}{2 \text{ msec/report}} = 15 \text{ reports/scan}$$

With a capability of servicing so many targets, it is expected that the queue will almost always be empty. Implementation of the M/D/1/n queue analysis verifies this as shown in the results of Table 6.

Table 6 - Display queue length

Queue size	Peak load	Nominal load
0	2.4×10^{-3}	5.9×10^{-8}
1	9.6×10^{-4}	1.2×10^{-14}

With little or no risk of overflow due to the short service time, the display queue need only be large enough to hold the data upon receipt; thereby, acting more like a buffer than a queue. For this reason, it is suggested that a queue size of 1 would be used for the new system.

Modem Queue Analysis

Based on the provided specifications, the maximum number of targets that can be handled by the modem per elevation scan is given by:

$$\frac{\text{sweep time}}{\text{service time}}\ 200 \text{ seg} = \frac{0.03 \text{ sec}}{\frac{10 \text{ bytes}}{1200 \text{ bytes/sec}} + 0.0007 \text{ sec}}\ 200 \text{ seg} = 664.2 \text{ reports/scan}$$

Seeing as though 664 is less than the specified 1400 peak load, one can conclude that the modem does not have the bandwidth to handle peak conditions and a priority queuing system is necessary to control the flow of data to the modem. The previous system had two priority filters whose azimuth gates are not allowed to overlap; the same assumption will be made for this analysis. A high priority tag is attached to any target lying within these two azimuth gates defined by the two priority filters.

Using the queuing analysis defined in the appendix, an imbedded Markov chain representing the scenario where the modem removes three reports per sweep time can be constructed. Obviously, if the average number of arriving targets is greater than 3, then the queue will always overflow. Therefore, for the purpose of design, the 85% modem capacity scenario was used as a basis for determining the modem queue size. For this case $\lambda = 565/200 = 2.82$ and an overflow probability of no more than one half of one percent were used as input parameters. The M/D/1/24 analysis provided a steady state vector, v^T, whose last term was approximately 0.0044, demonstrating that a queue size of 24 will meet the specifications. The full steady state vector can be found in the annex.

The low priority queue is effectively unanalyzable, but for this analysis, its safe to guess that the queue will be made large enough to hold

one azimuth sweep's worth of data. This should allow the operator enough time to respond to any alarms and change the size of his priority sectors.

Pseudo-code for Communication Processing

The pseudo-code below describes the processing required by the COMM CPU. All target reports for a given elevation scan are read in from GM and then passed on to the display queue. Then the report priority is checked. Low priority reports are placed into the low priority queue. High priority reports have their parameters (A_j, N_j, and Segment number) entered into a list of reports awaiting transfer to the high priority queue. In the pseudo-code, a high level IF statement is used to check for both priority filters. For this reason, 2*(3.40) is used as a execution time for that line of code. This approach ensures that no low priority reports go ahead of high priority reports that had been backed up or allow for high priority reports to be transferred non sequentially (when short high priority reports go ahead on long ones that had been backed up).

High priority target processing consists of a loop which sends as many elevation scans worth of data as possible. The loop stops whenever there is not sufficient room in the queue for the entire scan or until there are no longer any backup reports. To keep track of the backed up reports, head and tail pointers are used in conjunction with the array containing the detection report parameters.

Once a segment has been processed, irregardless of whether or not it was transferred to the modem queue, its respective semaphore is set to 0 so that RC will be able to bring in data during the next azimuth sweep. This will cause data that is not transmitted to the modem queue within 6 seconds to be lost.

In this code, 0.50 is used for an occurrence probability for deciding between low or high priority paths. In practice this will depend on the size of the high priority azimuth sectors as set by the operator.

The COMM pseudo-code does not do any reordering to handle the "corner turning problem".

pseudo-code	comment	execution time[8]
FOR (index = 1; index <= N_j; index++) {	/* GM to COMM local */	2.0 + 3.40*N_j
comm.report(index) = gm.report(index);	/* GM to COMM local */	(0.5+10/500)*1000*N_j
}		
FOR (index = 1; index <= N_j; index++) {	/* GM to COMM local */	2.0 + 3.40*N_j

[8]Total execution time based on the probability of being in high priority sector = probability of being in low priority = 0.5.

```
                                        /* COMM local to Display*/
   gm.display_q(index)  =  comm.report(index);          (0.7+35/200)*1000*Nj

IF (not in either high priority sector ) {                          2.0 * 3.40
                                        /* GM to COMM local */   0.5*(2.0 + 3.40*Nj)
   FOR (index = 1; index <= Nj; index++) {
                                        /* COMM local to Modem*/
      gm.modem_q0(index) = comm.range(index) ;      0.5*(0.7+10/1.2)*1000*Nj
   }
}
ELSE   /* high priority */
   IF (backup.seg(tail) = j);                                       0.5*3.40
      tail++;                                                       0.5*3.40
   backup.az(head) = Aj;                /* GM to COMM local */     0.5*5.40
   backup.n(head)= Nj;                  /* GM to COMM local */     0.5*5.40
   backup.seg(head)= j;                 /* COMM to COMM */         0.5*3.40
   head++;                                                          0.5*3.40
                                        /* GM to COMM local */     0.5*5.40
   DO WHILE (head <> tail)                                          0.5*3.40
      IF  (backup.n(tail) < number in modem queue)  {
                                        /* GM to COMM local */   0.5*(2.0 + 3.40*Nj)
         FOR (index = 1; index <= backup.n(tail); index++)
                                        /* COMM local to Modem*/
            gm.modem_q1(index) = comm.range(index);   0.5*(0.7+10/1.2)*1000*Nj
         tail++;                                                    0.5*3.40
      }
      ELSE break;                                                   3.40
   }
}
semaphorej=0;                           /* COMM to GM */           5.40
j++;   (modulo 200)                                                 3.40
                                                                  ----------
                                                                  10439Nj + 40.6
```

Timing Requirements

CPU load for this processor is made up of target table access, target load background, background diagnostics and operator requests for a change in the high priority azimuth sectors. Diagnostic processing for this CPU not only consists of self diagnostics, but also the receipt of the consolidated report from TP for transfer to Display. As is the case with the other CPUs, the variable load time is associated with the target report thread. Table 7 shows the execution time as a function of the number of targets and is calculated by:

target report processing = 10439(N) + 40.6 (μsec)

Table 7 - COMM report processing time

Number of Reports	Exec. time (msec)	P{list length=N}	Number of Reports	Exec. time (msec)	P{list length=N}
0	0.0406	5.94E-02	8	83.51	5.94E-03
1	10.44	1.68E-01	9	93.95	1.86E-03
2	20.88	2.37E-01	10	104.39	5.26E-04
3	31.32	2.23E-01	11	114.83	1.35E-04
4	41.76	1.57E-01	12	125.27	3.18E-05
5	52.20	8.88E-02	13	135.71	6.90E-06
6	62.63	4.18E-02	14	146.15	1.39E-06
7	73.07	1.68E-02	15	156.59	2.62E-07

Figure 9 - COMM processing tim

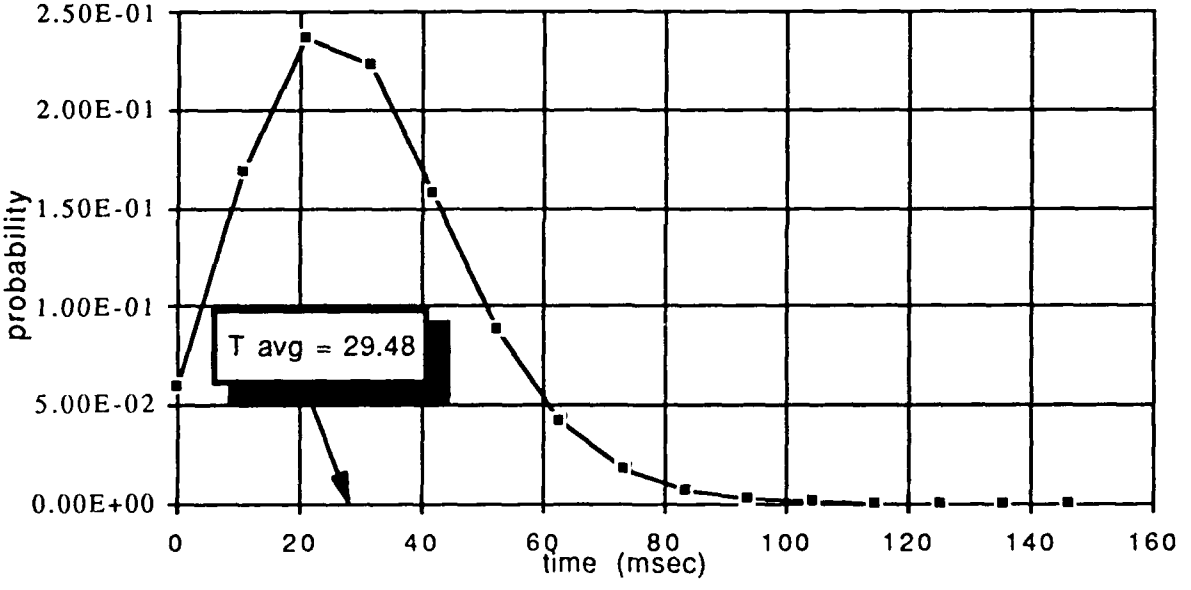


Figure 9 shows that even with the 565 target load, a 40% spare (on average) cannot be realized for this CPU. To maintain the 40% specification, a mean list length of 1.72 (344 targets) must be achieved.

$18 - (10.44N + 0.0406) = 0 \Rightarrow N = 1.72$ 60% utilization
$30 - (10.44N + 0.0406) = 0 \Rightarrow N = 2.87$ 100% utilization

Further more, with average of only 2.87 reports per elevation scan (574 targets in coverage), the COMM CPU will not be able to keep pace with the radar and high priority reports will be lost. These figures show that COMM is capable of handling the 85% modem capacity case were 565 targets are expected to be in coverage, but cannot do this and maintain the specified

50% spare. Given this condition, we will forgo the specifications on allowable CPU spare and determine expected response times to operator requests. Later in the report, a proposed solution to speeding up COMM will be addressed.

The probability of either a less than 40% spare or 0% spare for this processor under peak conditions are alarmingly high.

$$P(>\ 1 \text{ targets inside sector}) = 1 - \sum_{j=0}^{1} P(j \text{ targets})$$

$$= 0.993$$

$$P(>\ 2 \text{ targets inside sector}) = 1 - \sum_{j=0}^{2} P(j \text{ targets})$$

$$= 0.904$$

These numbers show that the system operator must keep tight controls on this priority filters so that the number of reports sent to the modem is kept below an allowable threshold, say 300 to 400 reports.

During nominal load conditions, operator requests for diagnostic reports and requests for new azimuth gates are processed Required processing times for the 344 target load are:

report processing = 18.00 msec

$$\text{no target load background} = \frac{150 \text{ msec}}{200 \text{ seg}} = 0.75 \text{ msec}$$

$$\text{background diagnostics} = \frac{3 \text{ sec}}{10 \text{ scans}} = 1.5 \text{ msec}$$

For COMM, it is now assumed that a 25% spare is acceptable for nominal processing.

remaining CPU time = (30 * 0.75) - (18.00 + 0.75 + 1.5) = 2.25 msec

and

diagnostic report processing = get operator request + send report to TP + receive 5 reports from TP + send 5 reports to Display

$$= \left(0.5 + \frac{5}{200}\right) + \left(0.5 + \frac{250}{200}\right) + \left(0.5 + \frac{5*250}{200}\right) + \left(0.7 + \frac{5*250}{200}\right)$$

$$= 0.525 + 1.75 + 6.75 + 6.75 = 15.775 \text{ msec}$$

If 2.13 msec is used to handle the operator request for a diagnostic report, then the expected response time to send a composite report to Display

(discounting any time spent waiting for the reports to arrive from TP) would be:

$$30.0\left(\frac{15.775}{2.25}\right) = 210.3 \text{ msec}$$

The processing load incurred as a result of a request to change a filter would consist of approximately 5 operator commands (1 request to change filter and min and max azimuth for each of two filters) plus a nominal overhead time (e.g., 0.7 msec). This total time would be 5(0.525) + 0.7 = 3.325 msec and can therefore expect to be accomplished within two elevation scans.

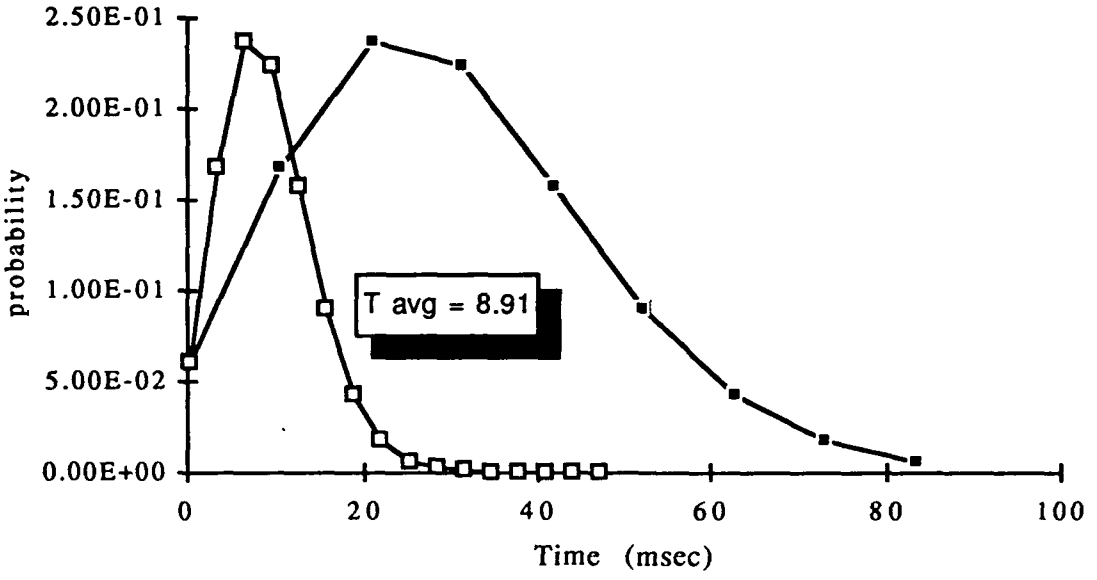
Timing analysis of this processor has shown that when the average list length is greater than 3.5, then COMM will not be able to perform its job within a 6 second azimuth scan. In these cases, the semaphores allowing RC to bring in new data will not have been reset and entire scans will be lost. One possible solution to assume that the new system would contain a software restructuring for the COMM processing, so that Low Priority targets are not sent to a low priority queue (a time consuming function), but instead have their address and list lengths stored similar to the way that high priority targets are backed up. This may reduce the risk of loosing some high priority data prior to the operator's changing of the azimuth gates, but would eventually lead to similar execution times after the change has gone into effect. (additional overhead to handle high priority targets vis a vis low priority is negligible).

A more likely change that would have a great impact of execution time would be the utilization of a new modem with an expanded bandwidth. If a 9600 baud machine were used in place of the specified 1200, execution time would now be defined by:

$$3146.9\ (N) + 40.6$$

Figure 10 shows the COMM execution time with the new modem vs that of the specified 1200 baud. The response of this new design on the order of the 8.91 msec average associated with the RC processor. Analysis of this subsystem indicates that the upgraded radar system must also be equipped with a more functional modem.

Figure 10 - COMM with 9600 baud vs 1200 baud



Bus Loading

The data transfer time is due to activity on each of the buses must be held under 30.0 msec per elevation scan. Each processor is individually addressed to examine the amount of time required to put data *onto* each of the two system buses.

Calculations are made using the 85% modem load (i.e., N=2.82 targets/elevation scan) except for calculations involving the COMM processor where the 344 target load was used (N=1.72) since this was shown to be a suitable load to handle the restricted bandwidth associated with the modem.

BUS 1

from RC = template to SP

$$= \left(0.3 + \frac{2000}{500}\right) = 4.3 \text{ msec/elevation scan}$$

from SP = target reports to RC + diagnostic report to RC

$$= \left(\left(0.3 + \frac{10}{500}\right) * N\right) + \left(0.3 + \frac{250}{500}\right)$$

$$= 0.9 + 0.8 \text{ msec/elevation scan}$$

total load on BUS1/elevation scan = 4.3 + 0.9 + 0.8 = 6.00 msec

BUS 2

from GM = template to RC + target data to TP + target reports to COMM

$$= \left(\frac{3.05 \text{ msec/seg}^{9}}{0.5 + \frac{2000}{200}}\right) + 2\,(0.0054)\,N + \left(\left(0.5 + \frac{10}{200}\right) * N\right)$$

$$= 0.29 + 0.03 + 1.55 = 1.87 \text{ msec/elevation scan}$$

from RC = target reports to GM + 2 diagnostic reports to TP

$$= \left(\left(0.5 + \frac{10}{200}\right) * N\right) + 2*\left(0.5 + \frac{250}{200}\right)$$

$$= 1.55 + 3.5 = 5.05 \text{ msec/elevation scan}$$

from TP = diagnostic reports to COMM

$$= \left(\frac{0.18 \text{ msec/seg}^{10}}{0.5 + \frac{5*250}{200}}\right)$$

$$= 0.03 \text{ msec/elevation scan}$$

from COMM = diagnostic report to TP + consolidated diag. report to DISPLAY + target reports to DISPLAY + target reports to MODEM

$$= \left(\frac{0.25 \text{ msec/seg}^{11}}{0.5 + \frac{5*250}{200}}\right) + \left(\frac{0.99 \text{ msec/seg}^{12}}{0.7 + \frac{250}{200}}\right) + \left(\left(0.7 + \frac{35}{200}\right) * N\right) + \left(\left(0.7 + \frac{10}{1.2}\right) * N\right)$$

$$= 0.14 + 0.14 + 1.51 + 15.54 = 17.33 \text{ msec/elevation scan}$$

from OIF = 3*diagnostic cmmds + template change cmd + change priority filter cmmd + display cmmd + diagnostic report

$$= 8 * \left(0.5 + \frac{5}{200}\right) + \left(0.5 + \frac{250}{200}\right)$$

$$= 4.20 + 1.75 = 5.95 \text{ msec}$$

total load on BUS2/elevation scan = 1.87 + 5.05 + 0.03 + 17.33 + 5.95 = 30.23 msec

---

[9] recalling that 6.1 msec per elevation scan is used to process the request for a new template; 3.27 is the proportional amount of that 6.1 msec used for accessing the global memory.

[10] during nominal operation 5.73 msec per scan is available for TP to process operator requests; 0.18 msec is the proportional amount of that 5.73 used for sending out the composite diagnostic report.

[11] during a 344 target load (25% spare) 2.25 msec per scan is available for COMM to process operator requests; 0.25 msec is the proportional amount of that 2.25 used for sending out the diagnostic report to TP

[12] during a 344 target load (25% spare) 2.25 msec per scan is available for COMM to process operator requests; 0.99 msec is the proportional amount of that 2.25 used for sending out the composite diagnostic report to the display.

The total load on BUS2 is dangerously close to the 30 msec scan time, but these calculation were made with all operator request going on at one time. In practice, only one request will be on going at one time, the most timely of which is the request for diagnostic reports. With the other commands removed, the total bus load is 27.32 msec.

Summary of Analysis Results

1. CPU loads for each computer under nominal and peak target loads.

   Detailed discussions of CPU timing and load requirements are addressed as each CPU is addressed above. These findings are summarized below:

Table 8 - Summary of CPU loads

	Radar Control	Target Processing	Communication
Nominal Exec. Time (msec)	3.1	7.27	36.6 cannot handle
Nominal Exec Time w/Operator Requests	15	15	cannot service requests
50% spare met (for average list length)	yes	yes	no
Peak Exec Time (msec)	7.1	16.72	83.52
Peak Exec. Time w/Operator Requests	18	18	cannot service requests
40% spare met (for average list length)	yes	yes	no
85% Modem Exec. Time (msec)	2.5	5.90	29.48
85% Modem Exec Time w/Operator Requests	15	15	cannot service requests

2. The expected delay experienced for targets measured from insertion in the signal processor queue to transmission over the modem lines, using the 85% modem load would be:

   SP Queue Wait Time + RC + TP + COMM + Modem Queue Wait Time =

   $$0.36 + 6.81 + 5.90 + 29.48 + 10.7 = 53.25 \text{ msec}$$

3. The probability of discarding one or more targets from a single beam.

   By specification, a maximum of eight targets can be recorded for each beam, therefore the probability of discarding one or more is found by Poisson approximation with N = number of targets in coverages including false targets p is:

$$p = \left(\frac{1 \text{ scan}}{200 \text{ segments}}\right)\left(\frac{1 \text{ segment}}{5 \text{ beams}}\right) = 3.33 \times 10^{-4}$$

$$P(> 8 \text{ targets inside sector}) = 1 - \sum_{j=0}^{1} P\{j \text{ targets}\}$$

$$= 1 - \sum_{j=0}^{8} e^{-0.533} \frac{(0.533)^j}{j!} \quad \text{at peak}$$

$$= 5.93 \times 10^{-9}$$

$$= 1 - \sum_{j=0}^{8} e^{-0.233} \frac{(0.233)^j}{j!} \quad \text{at nominal}$$

$$= \text{near zero}$$

$$= 1 - \sum_{j=0}^{8} e^{-0.188} \frac{(0.188)^j}{j!} \quad \text{at 85\% modem capacity}$$

$$= \text{near zero}$$

4. Queue sizes, for each of the data system queues, which is just adequate to meet the data system probability of loss specification.

Table 9 - Summary of Queue Analysis

	Signal Processing Queue	Display Queue	Modem Queue
Length	3	1	24
P {overflow}	0.0022	0.00094	0.0044
Specification Designed for	Peak Load	Peak Load	85% modem capacity

Final Thoughts

Detailed systems analysis, in excess of what has been presented here, must be performed in order to maintain the highest attainable level of knowledge concerning current and, in this case, projected foreign

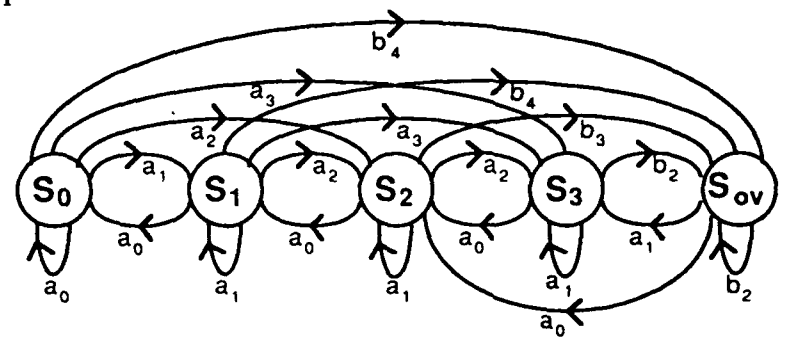
aerospace technology trends and capabilities. In addition to supporting research and development, these analyses allow for the assessment of performance capabilities, limitations, and vulnerabilities of current and future foreign weapon systems thereby precluding technological surprise to the United States and its allies.

In addition to analysis of an individual system, this type of work has application to large scale military engagement models. The U.S. Air Force is active in the use of software modeling and simulation techniques for purpose of wargaming. An example of this is a software tool called the Strategic Penetration Model (STRAPEM) developed by the Boeing Corporation. The STRAPEM code simulates the penetration of several hundred bomber aircraft, their accompanying escorts as well as any enemy interceptor aircraft they may enter the engagement. Queue size and probability of target loss analysis of the type provided in this report can be employed to control the dimensioning of data structures used in conjunction with the module responsible for modeling the radar. Application of these ideas will be addressed in future work.

Addendum

M/D/1/n Queue Analysis

The process necessary for determining a required queue size for an M/D/1/n[13] queue follows directly from the M/D/1 analysis presented by Trivedi[14] where an infinitely dimensional transition probability matrix is identified because the incoming job queue is infinite. For the M/D/1/3 case (represented by the Markov chain in figure A-1), the **P** matrix contains a right-most column which pulls together into one sink the infinitely long rows defined by Trivedi. It is from this example that M/D/1/n queues are discussed.



(Fig A-1)

$$\mathbf{P}=\begin{bmatrix} a_0 & a_1 & a_2 & a_3 & b_4 \\ a_0 & a_1 & a_2 & a_3 & b_4 \\ 0 & a_0 & a_1 & a_2 & b_3 \\ 0 & 0 & a_0 & a_1 & b_2 \\ 0 & 0 & a_0 & a_1 & b_2 \end{bmatrix} \quad \mathbf{V}^T=\mathbf{V}^T\mathbf{P} \Rightarrow \begin{cases} v_0a_0+v_1a_0=v_0 \\ v_0a_1+v_1a_1+v_2a_0=v_1 \\ v_0a_2+v_1a_2+v_2a_1+v_3a_0+v_{ov}a_0=v_2 \\ v_0a_3+v_1a_3+v_2a_2+v_3a_1+v_{ov}a_1=v_3 \\ v_0b_4+v_1b_4+v_2b_3+v_3b_2+v_{ov}b_2=v_{ov} \end{cases} \quad \text{(eq. A-1)}$$

where:

$$a_j = \text{prob}\{j \text{ items arrive for service in } (t_k, t_k + T_s)\} = e^{-\lambda Ts}\frac{(\lambda T_s)^j}{j!}$$

and

$$b_k = \text{prob}\{k \text{ or more items arrive for service in } (t_k, t_k + T_s)\} = 1-\sum_{j=0}^{k-1} a_j.$$

In addition to the n+1 equations generated by $V^T=V^TP$, the equation $\sum v_j=1$ must also hold since the system must only be in one of the n+1 possible system states. It would now appear that we have a situation

---

[13]This standard naming convention refers to a queue whose arrivals are exponentially distributed (M), constant service time (D), one server (1) and a queue length of n.

[14]Trivedi, Kishor Shridharbhai, Probability and Statistics with Reliability, Queuing, and Computer Science Applications, p.336-341.

where we are required to solve n+2 equations against n+1 unknowns (i.e., v_0 thru v_3); thus, indicating that one of these equations has been represented as a combination of another. This is the case and is easily identified by the fact that the first two rows of P are identical. This can also be realized by noticing that the determinate of the P matrix is equal to 0. We can now arbitrarily remove one of these equations and choose to eliminate the first ($v_0a_0+v_1a_0=v_0$). If all of the equations in (eq. A-1) above are rearranged by setting them equal to 0.

$$v_0a_1+v_1a_1+v_2a_0=v_1 \quad \Rightarrow \quad v_0a_1+v_1(a_1-1)+v_2a_0=0$$

The full set of n+1 equations can now be represented by the matrix equation $\mathbf{RV=S}$ where $\mathbf{R}$, $\mathbf{V}$, and $\mathbf{S}$ are the matrices shown.

$$\mathbf{R}=\begin{bmatrix} a_1 & a_1-1 & a_0 & 0 & 0 \\ a_2 & a_2 & a_1-1 & a_0 & a_0 \\ a_3 & a_3 & a_2 & a_1-1 & a_1 \\ b_4 & b_4 & b_3 & b_2 & b_2-1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \mathbf{V}=\begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} \quad \text{and} \quad \mathbf{S}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

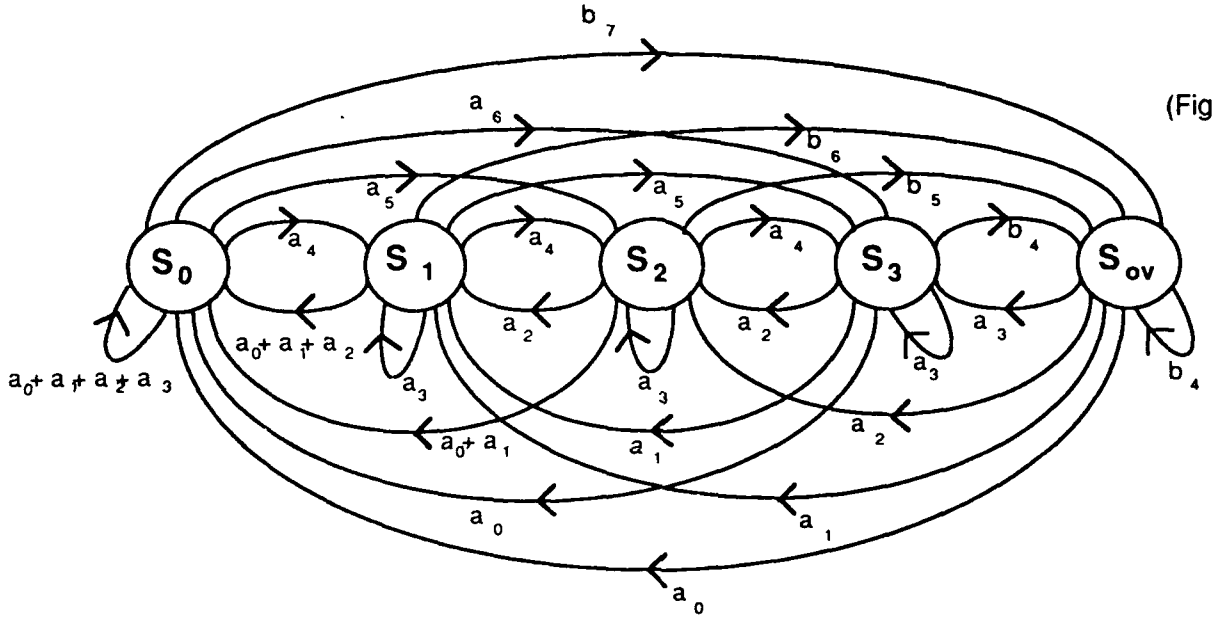
Solving for $\mathbf{V}$ by $\mathbf{V=R^{-1}S}$ would provide a set of steady state probabilities which represent the likelihood that the system would be in any particular state as time becomes infinitely large. Of particular interest in this case is v_4 which represents the probability that the system has been overflowed.

For the purpose of this project, as is typically the scenario in system design, the queue size is not pre-determined (instead, a specification of probability of overflow is given) and is therefore left as a design consideration for the engineer. To perform this analysis, the general case where P, R shown below are created until a value for n is found that meets the given overflow specification so that v_n is less than or equal to the specified allowable overflow. Once this has been achieved, *the required queue size is n-1* which is represented by the saturated state, S_{n-1}

$$\mathbf{P}=\begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} & b_n \\ a_0 & a_1 & a_2 & \cdots & a_n-1 & b_n \\ 0 & a_0 & a_1 & \cdots & a_{n-2} & b_{n-1} \\ 0 & 0 & a_0 & \cdots & a_{n-3} & b_{n-2} \\ & & & & & \\ 0 & \cdots & 0 & a_0 & 0 & b_1 \end{bmatrix} \quad \mathbf{R}=\begin{bmatrix} a_1 & a_1-1 & a_0 & 0 & \cdots & 0 \\ a_2 & a_2 & a_1-1 & a_0 & \cdots & 0 \\ a_{n-2} & a_{n-2} & a_{n-3} & \cdots & a_0 & a_0 \\ a_{n-1} & a_{n-1} & a_{n-2} & \cdots & a_1-1 & a_1 \\ b_n & b_n & b_{n-1} & \cdots & b_2 & b_2-1 \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{bmatrix}$$

The COMM Modem Queue and COMM Display Queue analysis is a slightly different to the SP queue analysis defined above. In the case of the COMM queues, a predefined number of reports will be handled before the COMM processor looks at the queue again. As described before, for the Modem queue, 3 reports will be handled before COMM sends more data and for the display it is 15.

For the purpose of describing the behavior of this queuing analysis, the imbedded Markov chain in Figure A-2 will be used. In this system as many as three reports can be handled by the receiving system (much like the modem) before new arrivals occur. It should be noted that this example implements a queue size of 3.



(Fig A-2)

The stochastic matrix, **P**, can be taken directly from this graph with a_i and b_i defined as before.

$$\mathbf{P}=\begin{bmatrix} a_0+a_1+a_2+a_3 & a_4 & a_5 & a_6 & b_7 \\ a_0+a_1+a_2 & a_3 & a_4 & a_5 & b_6 \\ a_0+a_1 & a_2 & a_3 & a_4 & b_5 \\ a_0 & a_1 & a_2 & a_3 & b_4 \\ a_0 & a_1 & a_2 & a_3 & b_4 \end{bmatrix}$$

$$\mathbf{V}^T=\mathbf{V}^T\mathbf{P}\Rightarrow\begin{cases} v_0(a_0+a_1+a_2+a_3)+v_1(a_0+a_1+a_2)+v_2(a_0+a_1)+v_3a_0+v_{ov}a_0=v_0 \\ v_0a_4+v_1a_3+v_2a_2+v_3a_1+v_{ov}a_1=v_1 \\ v_0a_5+v_1a_4+v_2a_3+v_3a_2+v_{ov}a_2=v_2 \\ v_0a_6+v_1a_5+v_2a_4+v_3a_3+v_{ov}a_3=v_3 \\ v_0b_7+v_1b_6+v_2b_5+v_3b_4+v_{ov}b_4=v_{ov} \end{cases} \quad \text{(eq. A-2)}$$

Again, $\sum v_j=1$ and one of the equations from eq. A-2 can be removed. Since the equation to be dropped is arbitrary, the first equation ($v_0(a_0+a_1+a_2+a_3)$ +

$v_1(a_0+a_1+a_2) + v_2(a_0+a_1) + v_3a_0 + v_{ov}a_0=v_0)$ being the most complicated would be the obvious choice. This results in an **R** matrix with the form:

$$\mathbf{R}=\begin{bmatrix} a_4 & a_3-1 & a_2 & a_1 & a_1 \\ a_5 & a_4 & a_3-1 & a_2 & a_2 \\ a_6 & a_5 & a_4 & a_3-1 & a_3 \\ b_7 & b_6 & b_5 & b_4 & b_4-1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad \mathbf{V}=\begin{bmatrix} v_0 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} \quad \text{and} \quad \mathbf{S}=\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

As opposed to developing a general form of **R** and **P** for this type of queue analysis, it should be observed that the SP queue is actually a special case of the COMM queues where the number guaranteed to be serviced before another arrival is 0. A more general form of **P** for queue analysis where 'g' is the number that can be taken from the queue before new data arrives and n is the queue size, is:

$$\mathbf{P}=\begin{bmatrix} \sum_{j=0}^{g} a_j & a_{g+1} & a_{g+2} & \cdots & a_{n+g} & b_{n+g+1} \\ \sum_{j=0}^{g-1} a_j & a_g & a_{g+1} & \cdots & a_{n+g-1} & b_{n+g} \\ \sum_{j=0}^{g-2} a_j & a_{g-1} & a_g & \cdots & a_{n+g-2} & b_{n+g-1} \\ a_0 & a_1 & a_2 & \cdots & a_g & b_{g+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{g-1} & a_g & b_{g+1} \end{bmatrix}$$

$$\mathbf{R}=\begin{bmatrix} a_{g+1} & a_g-1 & a_{g-1} & \cdots & a_0 & 0 \\ a_{g+2} & a_{g+1} & a_g & a_0 & \cdots & 0 \\ a_{n-2} & a_{n-2} & a_{n-3} & \cdots & a_0 & a_0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_n & b_n & b_{n-1} & \cdots & b_2 & b_2-1 \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{bmatrix}$$

P and **R** have dimension (n+1) x (n+1) where n is the queue size

Some points that may not be immediately obvious by observing the general form of P is that the summations in the first column is continued until G-I=0. Also there are a set of duplicate rows at the bottom of the matrix $[0,0,\ldots,a_{g-1},a_g,b_{g+1}]$ (as is the case for the bottom two rows of the stochastic matrix for the COMM queues).

The M/D/1/n program used to perform this analysis takes advantage of this general format. The code was written in LightSpeed C for the Macintosh. Information on the code can be attained from the author.

The code was not intended for operational use and therefore user interface coding received minimal attention. The three input parameters -; queue length, λt, and number of items guaranteed to be serviced before new entries arrive - are issued as #DEFINE statement entries at the beginning of the code; therefore it is necessary to edit and recompile the code for each run.

M/D/1/n Program Output

Signal Processor (SP) Queue under peak conditions

```
 Steady State analysis for M/D/1/3 queue
         lambda t = 0.240
   up to 0 items can be serviced prior to new arrivals

 the stochastic matrix P
7.87e-001   1.89e-001   2.27e-002   1.81e-003   1.14e-004
7.87e-001   1.89e-001   2.27e-002   1.81e-003   1.14e-004
0.00e+000   7.87e-001   1.89e-001   2.27e-002   1.93e-003
0.00e+000   0.00e+000   7.87e-001   1.89e-001   2.46e-002
0.00e+000   0.00e+000   7.87e-001   1.89e-001   2.46e-002

 the  matrix R
1.9e-001    -8.1e-001   7.9e-001    0.0e+000    0.0e+000
2.3e-002    2.3e-002    -8.1e-001   7.9e-001    7.9e-001
1.8e-003    1.8e-003    2.3e-002    -8.1e-001   1.9e-001
1.1e-004    1.1e-004    1.9e-003    2.5e-002    -9.8e-001
1.0e+000    1.0e+000    1.0e+000    1.0e+000    1.0e+000

 the steady state vector S
7.603e-001
2.062e-001
3.020e-002
3.061e-003
2.499e-004              probability of overflow
```

Modem Queue for a 9600 baud modem at peak conditions

```
 Steady State analysis for M/D/1/5 queue
         lambda t = 7.000
   up to 10 items can be serviced prior to new arrivals

 the stochastic matrix P
9.01e-001   4.52e-002   2.63e-002   1.42e-002   7.09e-003   3.31e-003   2.41e-003
8.30e-001   7.10e-002   4.52e-002   2.63e-002   1.42e-002   7.09e-003   5.72e-003
7.29e-001   1.01e-001   7.10e-002   4.52e-002   2.63e-002   1.42e-002   1.28e-002
5.99e-001   1.30e-001   1.01e-001   7.10e-002   4.52e-002   2.63e-002   2.70e-002
4.50e-001   1.49e-001   1.30e-001   1.01e-001   7.10e-002   4.52e-002   5.33e-002
3.01e-001   1.49e-001   1.49e-001   1.30e-001   1.01e-001   7.10e-002   9.85e-002
3.01e-001   1.49e-001   1.49e-001   1.30e-001   1.01e-001   7.10e-002   9.85e-002

 the  matrix R
4.5e-002    -9.3e-001   1.0e-001    1.3e-001    1.5e-001    1.5e-001    1.5e-001
2.6e-002    4.5e-002    -9.3e-001   1.0e-001    1.3e-001    1.5e-001    1.5e-001
1.4e-002    2.6e-002    4.5e-002    -9.3e-001   1.0e-001    1.3e-001    1.3e-001
7.1e-003    1.4e-002    2.6e-002    4.5e-002    -9.3e-001   1.0e-001    1.0e-001
3.3e-003    7.1e-003    1.4e-002    2.6e-002    4.5e-002    -9.3e-001   7.1e-002
2.4e-003    5.7e-003    1.3e-002    2.7e-002    5.3e-002    9.9e-002    -9.0e-001
1.0e+000    1.0e+000    1.0e+000    1.0e+000    1.0e+000    1.0e+000    1.0e+000
```

```
 the steady state vector S
8.755e-001
5.213e-002
3.257e-002
1.903e-002
1.046e-002
5.439e-003
4.914e-003              probability of overflow
```

Display Queue at peak conditions

```
 Steady State analysis for M/D/1/1 queue
      lambda t = 7.000
   up to 15 items can be serviced prior to new arrivals

 the stochastic matrix P
9.98e-001   1.45e-003   9.58e-004
9.94e-001   3.31e-003   2.41e-003
9.94e-001   3.31e-003   2.41e-003

 the  matrix R
1.4e-003    -1.0e+000   3.3e-003
9.6e-004    2.4e-003    -1.0e+000
1.0e+000    1.0e+000    1.0e+000

 the steady state vector S
9.976e-001
1.453e-003
9.617e-004              probability of overflow
```

MISSION

OF

ROME LABORATORY

Rome Laboratory plans and executes an interdisciplinary program in research, development, test, and technology transition in support of Air Force Command, Control, Communications and Intelligence (C^3I) activities for all Air Force platforms. It also executes selected acquisition programs in several areas of expertise. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C^3I systems. In addition, Rome Laboratory's technology supports other AFSC Product Divisions, the Air Force user community, and other DOD and non-DOD agencies. Rome Laboratory maintains technical competence and research programs in areas including, but not limited to, communications, command and control, battle management, intelligence information processing, computational sciences and software producibility, wide area surveillance/sensors, signal processing, solid state sciences, photonics, electromagnetic technology, superconductivity, and electronic reliability/maintainability and testability.